Amendments to the Claims

1. (Previously Presented): In multiprocessing, which is performed by using a plurality of processor modules, each of the plurality of processor modules including a cache memory and a plurality of processors sharing the cache memory, a method of allocating a task to each processor comprising the steps of:

monitoring access conditions of respective tasks to data shared among cache memories in the processor modules; and

allocating tasks that make frequent accesses to the same shared data to processors in the same module, on the basis of said access conditions.

2. (Original): The method according to claim 1, wherein said step of monitoring access conditions comprises the substeps of:

detecting an update to the shared data stored in a cache memory in one of the processor modules, said update causing an invalidation of the shared data stored in the cache memories in the other processor modules; and

storing identification information of tasks that caused said invalidation, and an address of the updated data, and the number of invalidations of the same data by the same task, and said step of allocating tasks comprises the substeps of:

classifying tasks into groups on the basis of the number of invalidations of the same data; and

allocating tasks in the same group to respective processors in the same processor module.

3. (Original): The method according to claim 1, wherein said step of monitoring access conditions comprises the substeps of:

detecting and storing an update to the shared data stored in one of the cache memories, said update causing an invalidation of the shared data stored in the other cache memories; and

monitoring access conditions of respective tasks to the invalidated data, and said step of allocating tasks comprises:

classifying tasks into groups on the basis of the number of invalidations of the same data; and

allocating tasks in the same group to respective processors in the same processor module.

4. (Original): The method according to claim 3, wherein said substep of monitoring access conditions comprises:

detecting an access to the invalidated data; and

storing an address of the invalidated data, identification information of a task that accessed the invalidated data, and the number of accesses to the same shared data by the same task.

5. (Original): The method according to claim 4, wherein said detecting an access to the invalidated data comprises: detecting an invalidation of data; storing an address of the invalidated data; detecting a cache miss; comparing an address of the data that caused the cache miss with the stored address of the invalidated data.

6. (Original): The method according to claim 2, further comprising the step of: making a request for allocation of tasks to an operating system performing multiprocessing, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

7. (Original): The method according to claim 4, further comprising the step of: making a request for allocation of tasks to an operating system performing multiprocessing, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

8. (Original): The method according to claim 5, further comprising the step of: making a request for allocation of tasks to an operating system performing multiprocessing, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

9. (Previously Presented): A multiprocessor system, in which a plurality of processor modules, each of the plurality of processor modules including one cache memory and a plurality of processors sharing the cache memory are used, comprising:

(a) a plurality of processor modules including: a detector for detecting accesses by respective tasks to data shared among cache memories in the processor modules; and a storage device for storing an address of the shared data, identification information of the tasks that accessed the shared data, and the number of accesses to the same shared data by the same task; and

(b) an allocator for allocating tasks that make frequent accesses to the same shared data to processors in the same module, on the basis of the number of accesses.

10. (Original): The multiprocessor system according to claim 9, wherein said detector for detecting accesses comprises:

a first device for detecting an update to the shared data stored in a cache memory in one of the processor modules, said update causing an invalidation of the shared data stored in the cache memories in the other processor modules; and

a second device for storing identification information of a task that caused said invalidation, and an address of the updated data, and the number of invalidations of the same data by the same task.

11. (Original): The multiprocessor system according to claim 9, wherein said detector for detecting accesses comprises:

a first device for detecting and storing an update to the shared data stored in one of a plurality of cache memories, said update causing an invalidation of the shared data stored in the other cache memories; and

an access detector for detecting accesses to the invalidated data by respective tasks.

12. (Original): The multiprocessor system according to claim 11, wherein said first device for detecting and storing an update to the shared data comprises: invalidation means for detecting an invalidation of data; and first address means for storing an address

of the invalidated data; and said access detector for detecting accesses comprises: cache detector means for detecting a cache miss; and second address means for comparing an address of the data that caused the cache miss with the stored address of the invalidated data.

13. (Original): The multiprocessor system according to claim 9, further comprising: request means for making a request for allocation of tasks to said allocator for allocating tasks, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

14. (Original): The multiprocessor system according to claim 10, further comprising: request means for making a request for allocation of tasks to said allocator for allocating tasks, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

15. (Previously Presented): A plurality of processor modules for use in a multiprocessor system, each of the plurality of processor modules including a cache memory and a plurality of processors sharing the cache memory, comprising:

a detector for detecting accesses by respective tasks to data shared among cache memories in the processor modules; and

a storage device for storing an address of the shared data, identification information of the tasks that accessed the shared data, and the number of accesses to the same shared data by the same task.

16. (Previously Presented): The plurality of processor modules according to claim 15, wherein said detector for detecting accesses comprises:

a first device for detecting an update to the shared data stored in a cache memory in one of the processor modules, said update causing an invalidation of the shared data stored in the cache memories in the other processor modules; and

a second device for storing identification information of a task that caused said invalidation, and an address of the updated data, and the number of invalidations of the

same data by the same task.

17. (Previously Presented): The plurality of processor modules according to claim 15, wherein said means for detecting accesses comprises:

a first device for detecting and storing an update to the shared data stored in one of the cache memories, said update causing an invalidation of the shared data stored in the other cache memories; and

an access detector for detecting accesses to the invalidated data by respective tasks.

18. (Previously Presented): The plurality of processor modules according to claim 17, wherein said means for detecting and storing an update to the shared data comprises:

invalidation means for detecting an invalidation of data; and first address means for storing an address of the invalidated data; and

said access detector for detecting accesses comprises:

cache detector means for detecting a cache miss; and

second address means for comparing an address of the data that caused the cache miss with the stored address of the invalidated data.

19. (Previously Presented): The plurality of processor modules according to claim 12, further comprising: request means for making a request for allocation of tasks to an operating system performing multiprocessing, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

20. (Previously Presented): The plurality of processor modules according to claim 16, further comprising: request means for making a request for allocation of tasks to an operating system performing multiprocessing, when the total number of said stored invalidations or said accesses exceeds a predetermined value.

21. (Currently Amended): A computer system using multiprocessors and a computer-readable recording medium embodying a program executable by the computer system, the program comprising:

(a) a plurality of processor modules including: a detector for detecting accesses by respective tasks to data shared among cache memories in the processor modules; and a storage device for storing an address of the shared data, identification information of the tasks that accessed the shared data, and the number of accesses to the same shared data by the same task; and

(b) an allocator for allocating a first set of tasks that make frequent accesses to ~~the same~~a first set of shared data to processors in ~~the same~~a first module of the plurality of processor modules, on the basis of the number of accesses, and for allocating a second set of tasks that make frequent accesses to a second set of shared data to processors in a second module of the plurality of processor modules, on the basis of the number of accesses;

wherein the tasks are allocated such that the number of accesses from different modules to the same shared data is minimized.